

## Resident Assembler Commands

F - FIRST PASS H - SECOND PASS OUTPUTS HEX CODE TO TAPE  
L - SECOND PASS OUTPUTS SOURCE LINE AND HEX EQUIVALENT

### Response to Initial ?

T ⇒ MAGNETIC TAPE

P ⇒ PAPER TAPE

### Utility Subroutines

#### (Standard Call and Return Conventions Apply)

READ 813E INPUT ASCII INTO RF.1  
READAH 813B SAME AS READ. IF HEX DIGIT, THEN  
DIGIT SHIFTED LEFT INTO RD  
TYPE6 81A2 OUTPUT ASCII CHAR AT M(R(6)). THEN INC R6  
TYPE 81A4 OUTPUT ASCII CHAR IN RF.1  
TYPE2 81AE OUTPUT 2 ASCII CHARS FROM HEX DIGIT PAIR  
IN RF.1  
TIMALC 80FE READ INPUT CHAR AND SET UP TIMING CONSTANT  
IN RE.1. INITIALIZE RC TO POINT TO DELAY1  
DELAY1 80EF DELAY AS A FUNCTION OF M(R(3)). THEN INC R3  
NOTE: ALL ROUTINES, EXCEPT DELAY1, USE P=3, EXIT  
WITH SEP R5 AND ALTER REGISTERS X, D, DF, RE, RF AND  
M(R(2))

EXAMPLE TO TYPE A LINE-FEED

EXAMPLE DELAY OF 1 BIT TIME (P=3)

```
LDI #0A    ..LOAD ASCII      SEP RC    ..CALL DELAY1
PHI RF     ..MOVE TO RF      #07      ..DELAY CONST.
SEP R4     ..CALL OUTPUT ROUTINE
,A (TYPE)
```

### ASCII (Without Parity)

|   | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9  | A   | B   | C  | D  | E  | F   |
|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|-----|-----|----|----|----|-----|
| 0 | NUL | SOH | STX | ETX | EOT | ENQ | ACK | BEL | BS  | HT | LF  | VT  | FF | CR | SO | SI  |
| 1 | DLE | DC1 | DC2 | DC3 | DC4 | NAK | SYN | ETB | CAN | EM | SUB | ESC | FS | GS | RS | US  |
| 2 | SP  | !   | "   | #   | \$  | %   | &   | '   | (   | )  | *   | +   | ,  | -  | .  | /   |
| 3 | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9  | :   | ;   | <  | =  | >  | ?   |
| 4 | @   | A   | B   | C   | D   | E   | F   | G   | H   | I  | J   | K   | L  | M  | N  | O   |
| 5 | P   | Q   | R   | S   | T   | U   | V   | W   | X   | Y  | Z   | [   | \  | ]  | ^  | _   |
| 6 |     | a   | b   | c   | d   | e   | f   | g   | h   | i  | j   | k   | l  | m  | n  | o   |
| 7 | p   | q   | r   | s   | t   | u   | v   | w   | x   | y  | z   | {   |    | }  | ~  | DEL |

EXAMPLES: A IS CODE 41. CARRIAGE-RETURN 0D

### Hexadecimal - Decimal Conversion

| HEX - BINARY | HEX - DEC | HEX - DEC | HEX - DEC | HEX - DEC |
|--------------|-----------|-----------|-----------|-----------|
| 0 0000       | 0 0       | 0 0       | 0 0       | 0 0       |
| 1 0001       | 1 4,096   | 1 256     | 1 16      | 1 1       |
| 2 0010       | 2 8,192   | 2 512     | 2 32      | 2 2       |
| 3 0011       | 3 12,288  | 3 768     | 3 48      | 3 3       |
| 4 0100       | 4 16,384  | 4 1,024   | 4 64      | 4 4       |
| 5 0101       | 5 20,480  | 5 1,280   | 5 80      | 5 5       |
| 6 0110       | 6 24,576  | 6 1,536   | 6 96      | 6 6       |
| 7 0111       | 7 28,672  | 7 1,792   | 7 112     | 7 7       |
| 8 1000       | 8 32,768  | 8 2,048   | 8 128     | 8 8       |
| 9 1001       | 9 36,864  | 9 2,304   | 9 144     | 9 9       |
| A 1010       | A 40,960  | A 2,560   | A 160     | A 10      |
| B 1011       | B 45,056  | B 2,816   | B 176     | B 11      |
| C 1100       | C 49,152  | C 3,072   | C 192     | C 12      |
| D 1101       | D 53,248  | D 3,328   | D 208     | D 13      |
| E 1110       | E 57,344  | E 3,584   | E 224     | E 14      |
| F 1111       | F 61,440  | F 3,840   | F 240     | F 15      |

# RCA 1800

MICROPROCESSOR

## Instruction Summary for the CDP1802 COSMAC Microprocessor

Information furnished by RCA is believed to be accurate and reliable. However, no responsibility is assumed by RCA for its use, nor for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent, patent rights of RCA.

MPM-920

Printed in USA 5/76

# CDP1802 Instruction Summary

| Op. Code | Mne-monic | Type      | Op. Code | Mne-monic | Type       | Op. Code | Mne-monic | Type       |
|----------|-----------|-----------|----------|-----------|------------|----------|-----------|------------|
| 00       | IDL       | Control   | 71       | DIS       | Control    | C8       | LSKP      | Skip       |
| 0N       | LDN       | Mem. Ref. | 72       | LDXA      | Mem. Ref.  |          | NLBR      | L-Branch   |
| 1N       | INC       | Reg. Op.  | 73       | STXD      | Mem. Ref.  | C9       | LBNO      | L-Branch   |
| 2N       | DEC       | Reg. Op.  | 74       | ADC       | Arith. Op. | CA       | LBNO      | L-Branch   |
| 30       | BR        | S-Branch  | 75       | SDB       | Arith. Op. | CB       | LBNF      | L-Branch   |
| 31       | BQ        | S-Branch  | 76       | SHRC      | Logic Op.  | CC       | LSIE      | Skip       |
| 32       | BZ        | S-Branch  | 77       | SMB       | Arith. Op. | CD       | LSQ       | Skip       |
| 33       | BDF       | S-Branch  | 78       | SAV       | Control    | CE       | LSZ       | Skip       |
| 34       | B1        | S-Branch  | 79       | MARK      | Control    | CF       | LSDF      | Skip       |
| 35       | B2        | S-Branch  | 7A       | REQ       | Control    | DN       | SEP       | Control    |
| 36       | B3        | S-Branch  | 7B       | SEQ       | Control    | EN       | SEX       | Control    |
| 37       | B4        | S-Branch  | 7C       | ADCI      | Arith. Op. | F0       | LDX       | Mem. Ref.  |
| 38       | NBR       | S-Branch  | 7D       | SDBI      | Arith. Op. | F1       | OR        | Logic Op.  |
|          | SKP       | Skip      | 7E       | SHLC      | Logic Op.  | F2       | AND       | Logic Op.  |
| 39       | BNQ       | S-Branch  | 7F       | SMBI      | Arith. Op. | F3       | XOR       | Logic Op.  |
| 3A       | BNZ       | S-Branch  | 8N       | GLO       | Reg. Op.   | F4       | ADD       | Arith. Op. |
| 3B       | BNF       | S-Branch  | 9N       | GHI       | Reg. Op.   | F5       | SD        | Arith. Op. |
| 3C       | BN1       | S-Branch  | AN       | PLO       | Reg. Op.   | F6       | SHR       | Logic Op.  |
| 3D       | BN2       | S-Branch  | BN       | PHI       | Reg. Op.   | F7       | SM        | Arith. Op. |
| 3E       | BN3       | S-Branch  | C0       | LBR       | L-Branch   | F8       | LDI       | Mem. Ref.  |
| 3F       | BN4       | S-Branch  | C1       | LBQ       | L-Branch   | F9       | ORI       | Logic Op.  |
| 4N       | LDA       | Mem. Ref. | C2       | LBZ       | L-Branch   | FA       | ANI       | Logic Op.  |
| 5N       | STR       | Mem. Ref. | C3       | LBDF      | L-Branch   | FB       | XRI       | Logic Op.  |
| 60       | IRX       | Reg. Op.  | C4       | NOP       | Control    | FC       | ADI       | Arith. Op. |
| 6N*      | OUT       | I/O       | C5       | LSNQ      | Skip       | FD       | SDI       | Arith. Op. |
| 6N*      | INP       | I/O       | C6       | LSNZ      | Skip       | FE       | SHL       | Logic Op.  |
| 70       | RET       | Control   | C7       | LSNF      | Skip       | FF       | SMI       | Arith. Op. |

\*61-67 ARE OUTPUT WHILE 69-6F ARE INPUT INSTRUCTIONS

## Utility Commands

!Maaa xx... CHANGE MEMORY AT aaaa TO xx...  
 ?Maaa hhhh LIST MEMORY AT aaaa FOR hhhh BYTES  
 \$Paaaa BEGIN PROGRAM EXECUTION AT aaaa WITH P=0

## Monitor Board Commands

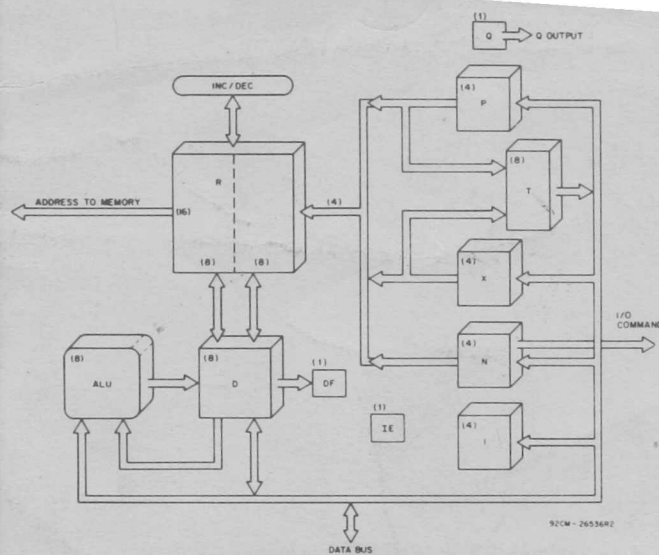
!Rn hhhh SET Rn TO hhhh      ?R      DISPLAY THE REGISTERS  
 !Xn      SET X TO n      ?X      DISPLAY X  
 !Pn      SET P TO n      ?P      DISPLAY P  
 !Dhh      SET D TO hh      ?D      DISPLAY D  
 !Fb      SET D FLAG TO b=0 OR 1      ?F      DISPLAY F  
 !BPaaaa SET A BREAKPOINT AT aaaa      \$P      RESUME PROGRAM  
 !BR      REMOVE THE BREAKPOINT      EXECUTION  
                                          SNhhhh EXECUTE THE NEXT hhhh  
                                          INSTRS.

NOTE: aaaa IS AN ADDRESS      n IS A REGISTER NUMBER  
       xx IS A HEX DIGIT PAIR      h IS A HEX DIGIT

## Resident Editor Commands

B      - MOVE TO BEGINNING      nX      - SAVE n LINES  
           OF BUFFER      G      - GET THE SAVED LINES  
 Z      - MOVE TO END OF BUFFER      nT      - TYPE n LINES  
 nC      - MOVE BY n CHARACTERS      nP      - OUTPUT n LINES  
 nL      - MOVE BY n LINES      nW      - OUTPUT AND DELETE THE  
                                          1<sup>st</sup> n LINES  
 A      - APPEND TO END OF BUFFER  
 nD      - DELETE n CHARACTERS      E      - END THE EDIT SESSION  
 nK      - DELETE n LINES      N      - OUTPUT 60 NULLS TO  
                                          PAPER TAPE  
 I text\$ - INSERT text

\$searchtext\$substitutetext\$ - FIND AND REPLACE THE TEXT      F text\$ - FIND text



| DF | 1 Bit   | Data Flag (DF)                               |
|----|---------|----------------------------------------------|
| R  | 16 Bits | 1 of 16 Scratchpad Registers                 |
| P  | 4 Bits  | Designates which register is Program Counter |
| X  | 4 Bits  | Designates which register is Data Pointer    |
| N  | 4 Bits  | Low-order Instruction Digit                  |
| I  | 4 Bits  | High-order Instruction Digit                 |
| T  | 8 Bits  | Holds old X, P after Interrupt               |
| IE | 1 Bit   | Interrupt Enable                             |
| Q  | 1 Bit   | Output Flip-flop                             |

Interrupt Action: X and P are stored in T after executing current instruction; designator P is set to 1; designator X is set to 2; interrupt enable is reset to 0 (inhibit); and the interrupt request is serviced.

DMA Action: Finish executing current instruction, R(0) points to memory area for data transfer; data is loaded into or read out of memory; and increment R(0).

Note: In the event of concurrent DMA and INTERRUPT requests, DMA has priority.

External Flags: Four one-bit Flags set externally and tested by some branching instructions.

### CDP1802 Microprocessor Instructions

| OP CODE                     | LEVEL1 SYNTAX | NAME               | ACTION                                             |
|-----------------------------|---------------|--------------------|----------------------------------------------------|
| <b>Control Instructions</b> |               |                    |                                                    |
| 00                          | IDL           | IDLE               | WAIT FOR DMA OR INTERRUPT; M(R(0)) → BUS           |
| C4                          | NOP           | NO OPERATION       | CONTINUE                                           |
| DN                          | SEP reg       | SET P              | N → P                                              |
| EN                          | SEX reg       | SET X              | N → X                                              |
| 7B                          | SEQ           | SET Q              | 1 → Q                                              |
| 7A                          | REQ           | RESET Q            | 0 → Q                                              |
| 78                          | SAV           | SAVE               | T → M(R(X))                                        |
| 79                          | MARK          | PUSH X, P TO STACK | (X, P) → T; (X, P) → M(R(2)); THEN P → X; R(2) - 1 |
| 70                          | RET           | RETURN             | M(R(X)) → (X, P); R(X) + 1; 1 → IE                 |
| 71                          | DIS           | DISABLE            | M(R(X)) → (X, P); R(X) + 1; 0 → IE                 |

### Memory Reference

|    |          |                           |                          |
|----|----------|---------------------------|--------------------------|
| 0N | LDN reg  | LOAD VIA N                | M(R(N)) → D; FOR N NOT 0 |
| 4N | LDA reg  | LOAD ADVANCE              | M(R(N)) → D; R(N) + 1    |
| F0 | LDX      | LOAD VIA X                | M(R(X)) → D              |
| 72 | LDXA     | LOAD VIA X AND ADVANCE    | M(R(X)) → D; R(X) + 1    |
| F8 | LDI expr | LOAD IMMEDIATE            | M(R(P)) → D; R(P) + 1    |
| 5N | STR reg  | STORE VIA N               | D → M(R(N))              |
| 73 | STXD     | STORE VIA X AND DECREMENT | D → M(R(X)); R(X) - 1    |

### Register Operations

|    |         |                 |            |
|----|---------|-----------------|------------|
| 1N | INC reg | INCREMENT REG N | R(N) + 1   |
| 2N | DEC reg | DECREMENT REG N | R(N) - 1   |
| 60 | IRX     | INCREMENT REG X | R(X) + 1   |
| 8N | GLO reg | GET LOW REG N   | R(N).0 → D |
| AN | PLO reg | PUT LOW REG N   | D → R(N).0 |
| 9N | GHI reg | GET HIGH REG N  | R(N).1 → D |
| BN | PHI reg | PUT HIGH REG N  | D → R(N).1 |

### Logic Operations\*\*

|    |          |                        |                                         |
|----|----------|------------------------|-----------------------------------------|
| F1 | OR       | OR                     | M(R(X)) OR D → D                        |
| F9 | ORI expr | OR IMMEDIATE           | M(R(P)) OR D → D; R(P) + 1              |
| F3 | XOR      | EXCLUSIVE OR           | M(R(X)) XOR D → D                       |
| FB | XRI expr | EXCLUSIVE OR IMMEDIATE | M(R(P)) XOR D → D; R(P) + 1             |
| F2 | AND      | AND                    | M(R(X)) AND D → D                       |
| FA | ANI expr | AND IMMEDIATE          | M(R(P)) AND D → D; R(P) + 1             |
| F6 | SHR      | SHIFT RIGHT            | SHIFT D RIGHT, LSB(D) → DF, 0 → MSB(D)  |
| 76 | *SHRC    | SHIFT RIGHT WITH CARRY | SHIFT D RIGHT, LSB(D) → DF, DF → MSB(D) |
|    | *RSHR    | RING SHIFT RIGHT       |                                         |
| FE | SHL      | SHIFT LEFT             | SHIFT D LEFT, MSB(D) → DF, 0 → LSB(D)   |
| 7E | *SHLC    | SHIFT LEFT WITH CARRY  | SHIFT D LEFT, MSB(D) → DF, DF → LSB(D)  |
|    | *RSHL    | RING SHIFT LEFT        |                                         |

### Arithmetic Operations\*\*

|    |           |                                        |                                          |
|----|-----------|----------------------------------------|------------------------------------------|
| F4 | ADD       | ADD                                    | M(R(X)) + D → DF, D                      |
| FC | ADI expr  | ADD IMMEDIATE                          | M(R(P)) + D → DF, D; R(P) + 1            |
| 74 | ADC       | ADD WITH CARRY                         | M(R(X)) + D + DF → DF, D;                |
| 7C | ADCI expr | ADD WITH CARRY, IMMEDIATE              | M(R(P)) + D + DF → DF, D; R(P) + 1       |
| F5 | SD        | SUBTRACT D                             | M(R(X)) - D → DF, D                      |
| FD | SDI expr  | SUBTRACT D IMMEDIATE                   | M(R(P)) - D → DF, D; R(P) + 1            |
| 75 | SDB       | SUBTRACT D WITH BORROW                 | M(R(X)) - D - (NOT DF) → DF, D           |
| 7D | SDBI expr | SUBTRACT D WITH BORROW IMMEDIATE       | M(R(P)) - D - (NOT DF) → DF, D; R(P) + 1 |
| F7 | SM        | SUBTRACT MEMORY                        | D - M(R(X)) → DF, D                      |
| FF | SMI expr  | SUBTRACT MEMORY IMMEDIATE              | D - M(R(P)) → DF, D; R(P) + 1            |
| 77 | SMB       | SUBTRACT MEMORY WITH BORROW            | D - M(R(X)) - (NOT DF) → DF, D           |
| 7F | SMBI expr | SUBTRACT MEMORY WITH BORROW, IMMEDIATE | D - M(R(P)) - (NOT DF) → DF, D; R(P) + 1 |

## Short-Branch

|    |      |      |                                  |                                                          |
|----|------|------|----------------------------------|----------------------------------------------------------|
| 30 | BR   | expr | SHORT BRANCH                     | $M(R(P)) \rightarrow R(P).0$                             |
| 38 | *NBR | expr | NO SHORT BRANCH (SEE SKP)        | $R(P) + 1$                                               |
| 32 | BZ   | expr | SHORT BRANCH IF D = 0            | IF D=0, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$     |
| 3A | BNZ  | expr | SHORT BRANCH IF D NOT 0          | IF D NOT 0, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$ |
| 33 | *BDF | expr | SHORT BRANCH IF DF=1             | IF DF=1, $M(R(P)) \rightarrow R(P).0$                    |
|    | *BPZ | expr | SHORT BRANCH IF POS OR ZERO      | ELSE $R(P) + 1$                                          |
|    | *BGE | expr | SHORT BRANCH IF GREATER OR EQUAL |                                                          |
| 3B | *BNF | expr | SHORT BRANCH IF DF=0             | IF DF=0, $M(R(P)) \rightarrow R(P).0$                    |
|    | BM   | expr | SHORT BRANCH IF MINUS            | ELSE $R(P) + 1$                                          |
|    | BL   | expr | SHORT BRANCH IF LESS             |                                                          |
| 31 | BQ   | expr | SHORT BRANCH IF Q=1              | IF Q=1, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$     |
| 39 | BNQ  | expr | SHORT BRANCH IF Q=0              | IF Q=0, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$     |
| 34 | B1   | expr | SHORT BRANCH IF EF1=1            | IF EF1=1, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$   |
| 3C | BN1  | expr | SHORT BRANCH IF EF1=0            | IF EF1=0, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$   |
| 35 | B2   | expr | SHORT BRANCH IF EF2=1            | IF EF2=1, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$   |
| 3D | BN2  | expr | SHORT BRANCH IF EF2=0            | IF EF2=0, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$   |
| 36 | B3   | expr | SHORT BRANCH IF EF3=1            | IF EF3=1, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$   |
| 3E | BN3  | expr | SHORT BRANCH IF EF3=0            | IF EF3=0, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$   |
| 37 | B4   | expr | SHORT BRANCH IF EF4=1            | IF EF4=1, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$   |
| 3F | BN4  | expr | SHORT BRANCH IF EF4=0            | IF EF4=0, $M(R(P)) \rightarrow R(P).0$ ELSE $R(P) + 1$   |

## Long Branch

|    |       |      |                           |                                                                                            |
|----|-------|------|---------------------------|--------------------------------------------------------------------------------------------|
| C0 | LBR   | expr | LONG BRANCH               | $M(R(P)) \rightarrow R(P).1; M(R(P) + 1) \rightarrow R(P).0$                               |
| C8 | *NLBR | expr | NO LONG BRANCH (SEE LSKP) | $R(P) + 2$                                                                                 |
| C2 | LBZ   | expr | LONG BRANCH IF D=0        | IF D=0, $M(R(P)) \rightarrow R(P).1; M(R(P) + 1) \rightarrow R(P).0$ ; ELSE $R(P) + 2$     |
| CA | LBNZ  | expr | LONG BRANCH IF D NOT 0    | IF D NOT 0, $M(R(P)) \rightarrow R(P).1; M(R(P) + 1) \rightarrow R(P).0$ ; ELSE $R(P) + 2$ |
| C3 | LBDF  | expr | LONG BRANCH IF DF=1       | IF DF=1, $M(R(P)) \rightarrow R(P).1; M(R(P) + 1) \rightarrow R(P).0$ ; ELSE $R(P) + 2$    |
| CB | LBNF  | expr | LONG BRANCH IF DF=0       | IF DF=0, $M(R(P)) \rightarrow R(P).1; M(R(P) + 1) \rightarrow R(P).0$ ; ELSE $R(P) + 2$    |
| C1 | LBQ   | expr | LONG BRANCH IF Q=1        | IF Q=1, $M(R(P)) \rightarrow R(P).1; M(R(P) + 1) \rightarrow R(P).0$ ; ELSE $R(P) + 2$     |
| C9 | LBNQ  | expr | LONG BRANCH IF Q=0        | IF Q=0, $M(R(P)) \rightarrow R(P).1; M(R(P) + 1) \rightarrow R(P).0$ ; ELSE $R(P) + 2$     |

## Skip Instructions

|    |       |                      |                                        |
|----|-------|----------------------|----------------------------------------|
| 38 | *SKP  | SHORT SKIP (SEE NBR) | $R(P) + 1$                             |
| C8 | *LSKP | LONG SKIP (SEE NLBR) | $R(P) + 2$                             |
| CE | LSZ   | LONG SKIP IF D=0     | IF D=0, $R(P) + 2$ ; ELSE CONTINUE     |
| C6 | LSNZ  | LONG SKIP IF D NOT 0 | IF D NOT 0, $R(P) + 2$ ; ELSE CONTINUE |
| CF | LSDF  | LONG SKIP IF DF=1    | IF DF=1, $R(P) + 2$ ; ELSE CONTINUE    |
| C7 | LSNF  | LONG SKIP IF DF=0    | IF DF=0, $R(P) + 2$ ; ELSE CONTINUE    |
| CD | LSQ   | LONG SKIP IF Q=1     | IF Q=1, $R(P) + 2$ ; ELSE CONTINUE     |
| C5 | LSNQ  | LONG SKIP IF Q=0     | IF Q=0, $R(P) + 2$ ; ELSE CONTINUE     |
| CC | LSIE  | LONG SKIP IF IE=1    | IF IE=1, $R(P) + 2$ ; ELSE CONTINUE    |

## Input-Output Byte Transfer

|    |     |     |        |                                                                                 |
|----|-----|-----|--------|---------------------------------------------------------------------------------|
| 6N | OUT | dev | OUTPUT | $M(R(X)) \rightarrow \text{BUS}; R(X) + 1; \text{FOR } N=1 \text{ TO } 7$       |
| 6N | INP | dev | INPUT  | $\text{BUS} \rightarrow M(R(X)); \text{BUS} + D; \text{FOR } N=9 \text{ TO } F$ |

### NOTE:

N A HEX DIGIT  
 reg A HEX DIGIT, "R" FOLLOWED BY A HEX DIGIT, OR A SYMBOLIC NAME.  
 dev "1" THROUGH "7" OR A SYMBOLIC NAME IN THAT RANGE.  
 expr A CONSTANT, "++", OR A SYMBOLIC NAME POSSIBLY PLUS ("++") OR MINUS ("--") A CONSTANT.  
 \* THIS INSTRUCTION IS ASSOCIATED WITH MORE THAN ONE MNEMONIC. EACH MNEMONIC IS INDIVIDUALLY LISTED.  
 \*\* THE ARITHMETIC OPERATIONS AND THE SHIFT INSTRUCTIONS ARE THE ONLY INSTRUCTIONS THAT CAN ALTER THE DF.  
 AFTER AN ADD INSTRUCTION:  
 DF=1 DENOTES A CARRY HAS OCCURRED  
 DF=0 DENOTES A CARRY HAS NOT OCCURRED  
 AFTER A SUBTRACT INSTRUCTION:  
 DF=1 DENOTES NO BORROW, D IS A TRUE POSITIVE NUMBER  
 DF=0 DENOTES A BORROW, D IS TWO'S COMPLEMENT  
 THE SYNTAX -- (NOT DF) DENOTES THE SUBTRACTION OF THE BORROW